TSX-Plus Version 5.1C

Release Notes

These release notes describe the differences between TSX-Plus version 5.1C and versions 5.1 and 5.1B. The TSX-Plus Reference Manual and TSX-Plus System Manager's Guide incorporate the information described in earlier release notes. In case of any differences between the release notes and the manuals, the release notes take precedence.

April, 1985

CONTENTS

TSX-Plus Version 5.1C Release Notes

1. Support for Professional 380

Support is now provided for the Professional 380 personal computer as well as the Professional 350. PRO/TSX-Plus can be used with RT-11 version 5.1C on the Pro-380 and with either RT-11 version 5.1B or 5.1C on the Pro-350. The DW.TSX handler provided with PRO/TSX-Plus version 5.1C will automatically determine the size of the Winchester disk. The installation procedure on the Pro is the same as with PRO/TSX-Plus version 5.1B.

2. Terminal character length and parity control

TSX-Plus version 5.1C provides support for controlling the number of bits and parity of transmitted and received characters. This control may be specified through system generation options, keyboard SET commands, and system service calls (EMT's). Character length may be 7 or 8 bits. Even, odd, or no parity may be selected. Character length and parity support may be used with lines connected to the following controller types: DZ11, DZV11, DH11, DHV11, Pro communications port, and the Pro printer port.

The SPEED macro in TSGEN has been modified to accept two additional optional arguments that specify the character length and parity setting. The new form of the SPEED macro is:

```
SPEED  speed[,length,parity]
```

where speed is the speed code, specified in the same form as before (e.g., S1200, S9600, etc.), length is 7. or 8. indicating the number of data bits in each character, and parity is EVEN, ODD, or NONE. If the length and parity parameters are omitted, they default to 8 bits, no parity. For example, the following SPEED macro specifies that the line is to operate at 9600 baud, with 7 bit characters and even parity.

```
SPEED  S9600,7.,EVEN
```

The following SPEED macro specifies 9600 baud, 8 bit characters, and no parity.

```
SPEED  S9600,8.,NONE
```

Two new qualifiers may be used with the SET TERMINAL command to specify character length and parity. The qualifier used to specify character length is BITS=n where n is either 7 or 8. The qualifier used to specify the parity is PARITY={EVEN ODD NONE}. PARITY=NONE may be alternatively specified as NOPARITY. For example, the following command specifies that line 3 should use 7 bit characters with even parity.

```
SET TERM 3 BITS=7,PARITY=EVEN
```

The following command selects 8 bit characters with no parity for the current terminal line.

```
SET TERM BITS=8,NOPARITY
```

There are now two qualifiers for the SET TERMINAL command related to character length: [NO]EIGHTBIT and BITS=n. These qualifiers have different functions. The [NO]EIGHTBIT qualifier provides software control over the number of bits retained in each character processed by the TSX-Plus terminal control module. If NOEIGHTBIT is specified, each character is masked to 7 bits before being processed and passed to the user's program. The BITS=n qualifier provides hardware control. It causes TSX-Plus to set the specified character length into the hardware line controller for the line. Setting a value for one of these qualifiers does not affect the other. Thus it is possible to use 8 bit communication between the computer and a terminal but mask received characters to 7 bits before passing them to the program. In order to transmit and receive 8 bit characters it is necessary to set both the EIGHTBIT and BITS=8 qualifiers.

The PARITY and BITS qualifiers may also be used with the SET CL command to control these options for communication lines. For example, the following command sets CL1 to 7 bit characters with even parity.

```
SET CL1 BITS=7,PARITY=EVEN
```

If autobaud is specified for a line, that line is automatically set to 8 bit characters with no parity. Terminals must be set correspondingly if they are to be used with an autobaud line. Once a connection has been established to an autobaud line and the speed has been determined, the SET TERMINAL command can be used to change the character length and parity. The line will revert to 8 bits with no parity when the connection is broken and the speed becomes indeterminate.

The TSX-Plus EMT which sets line speed (function code 0,152), and the CL special function (function code 256) have been modified to allow character length and parity control to be specified along with line speed. The value that is used to specify the line parameters now has the following form (in binary):

```
OPLxSSSS
```

The low-order 4 bits "SSSS" specify the speed; the same speed codes are used as before. Bit 4 "x" is not used. Bit 5 "L" specifies the character length. If this bit is 0, the character length is 8 bits; if this bit is 1, the character length is 7 bits. Bit 6 "P" specifies if parity control is wanted. If this bit is 0, no parity is selected and bit 7 is ignored; if this bit is 1, parity generation and checking is enabled. Bit 7 "O" selects even or odd parity and is only meaningful if bit 6 is 1. If bit 7 is 0, even parity is selected; if bit 7 is 1, odd parity is selected. Note that if only the speed value is specified, with all other bits zero, 8 bit characters with no parity are selected.

The SHOW TERMINAL command can be used to display the number of bits per character and parity control status for each line. The character length and parity are displayed in the column with the terminal speed value on the SHOW TERMINAL listing. The SYSMON utility has been updated correspondingly.

3. PAGE qualifier for terminal control

The [NO]PAGE qualifier may be used with the SET TERMINAL command to select whether control-S (XOFF) and control-Q (XON) characters are intercepted by the operating system and used for flow control or passed to the program. If PAGE is specified, control-S and control-Q are processed by the system and cause transmission to be stopped and started. If NOPAGE is specified, control-S and control-Q are not interpreted by the system and are passed to the user's program like ordinary characters. The $PAGE sysgen parameter may be used to assign initial values for this parameter to each line using the FLAGS macro within line definition blocks. The $PAGE setting is automatically selected if any of the following terminal types are specified for a line: VT100, VT200, VT52, LA120, Diablo, Qume. The SET TERMINAL NOPAGE command can be used to override this setting after the terminal initialization has been done.

4. Qualifiers for SET TERMINAL and SET CL commands

Multiple qualifiers may now be specified on the SET TERMINAL and SET CL commands. If there is more than one qualifier the qualifiers may be separated by a space, a comma, or a slash. The following commands illustrate multiple qualifiers used with the same command.

```
SET TERM SPEED=9600,BITS=8,NOPARITY
SET CL2 SPEED=1200,BITS=7,PARITY=EVEN
```

5. Improvements to SYSMON Program

The SYSMON program may now be used with terminals that support the Hazeltine and ADM3A protocols as well as with VT52, VT100 and VT200 terminals. The position of jobs in memory is now provided as part of SYSMON screen 2 (job execution status). The position information is the number of K-bytes from the base of physical memory to the start of the job's memory area.

6. New CL handler special functions

Three new .SPFUN codes are now supported with the CL (communication line) facility.

Function 261 -- Determine number of input characters pending. One word is returned into the user buffer, containing the number of characters available to be read from the CL unit associated with the specified channel. This function can be used to test for pending input prior to issuing a read (.READx or .SPFUN) on the channel. If no characters are pending, attempts to read from the channel will not complete until the word (or byte) count is fulfilled. If you do not wish to have a read request pending until the word (or byte) count is fulfilled, then first determine the number of characters pending in the input buffer. If there are none, then do not issue the read.

Function 262 -- Determine the number of output characters pending. One word is returned into the user buffer, containing the number of characters in the output buffer which have not yet been transmitted.

Function 263 -- Write with byte count. This special function is used to write a block of characters to a CL line with the length of the block specified by a byte count rather than the word count used with the .WRITE EMT. This is useful in situations where an odd number of bytes must be written and null characters cannot be used to pad out the last word. The .SPFUN has the standard form except a byte count is specified for the fifth parameter which is normally used to specify a word count. For example, the following .SPFUN writes 21 bytes from a buffer named OUTBUF:

```
.SPFUN #AREA,#1,#263,#OUTBUF,#21.,#0
```

7. Account authorization file changes (TSAUTH and LOGON)

The TSAUTH and LOGON programs have been changed to improve system security. For example, user passwords are no longer displayed when using the "LIST" command in TSAUTH and are not echoed when authorizing a new account. If a user forgets his/her password, then the account should be listed, killed and reauthorized with a new password. You can no longer determine the password by listing the account with TSAUTH. Remember, it is still invalid to have duplicate account PPN´s and invalid to have the same user name for more than one PPN.

You must copy the new LOGON.SAV program to your system disk in order to use the new authorization mechanism. In addition, if you use the new TSAUTH program, you must use the new LOGON program. Old authorization files are automatically converted to the new method by either of the new TSAUTH or LOGON programs, so it is not necessary to initialize a new authorization file with all new accounts. In addition, for those using specialized charge accounting programs, there has been no change to the layout of the CHARGE file.

8. Increased Number of ASSIGN and ACCESS Commands

The number of ASSIGN commands that may be in effect for a job has been increased from 15 to 25. The maximum number of ACCESS commands for a job has been increased from 20 to 25.

9. New Sysgen Parameter for Unexpected Interrupts

A new parameter has been added to TSGEN to control the action of the system in response to an interrupt that occurs at an unexpected location. The UXIFLG parameter controls the action taken by TSX-Plus when an interrupt occurs at an unexpected location. Unexpected interrupts may occur if the interrupt vector address specified in a device handler does not match the actual interrupt address for which the device has been set. Unexpected interrupts can also occur if real-time interrupts occur and no connection has been established between the real-time interrupt and a TSX-Plus real-time program.

If UXIFLG is set to 1 (one) then unexpected interrupts cause a fatal system error with the error message:

```
?TSX-F-UEI-Interrupt occurred at unexpected location
Argument value = xxx
```

Where "xxx" is the vector through which the interrupt occurred.

If UXIFLG is set to 0 (zero) then unexpected interrupts are ignored by the system and do not cause a fatal system error or print an error message.

The recommended setting for UXIFLG is 1 (one).

10. Nesting of Completion and .SYNCH routines

A change has been made to the way completion routines are processed to allow some completion routines to interrupt other completion routines. In version 5.1C, there are three classes of completion routines. Normal completion routines such as for I/O completion, .MRKT, etc., are of class 1; completion routines scheduled as a result of real-time interrupts and as a result of executing the TSX-Plus real-time EMT with function code 21,140 are of class 2; completion routines for .SYNCH requests are of class 3. A completion request of a higher class may interrupt a currently-executing completion routine of a lower class. Thus, a normal completion routine could be interrupted by a real-time completion routine, which in turn could be interrupted by a .SYNCH completion routine. Completion routines cannot be interrupted by completion requests of the same or lower class; thus, completion requests of the same class are serialized.

11. BPT Instruction Can be Used to Enter Debugger

A BPT (breakpoint trap) instruction can now be used to trigger execution of the system debugging program. If a BPT instruction is executed by a program which is not currently associated with the system debugger and location 14 (trap vector for BPT) in the program is 0 (zero), then the system debugger is entered with the instruction that follows the BPT as the current instruction, assuming that system debugger support has been selected in TSGEN (DBGFLG=1).

12. Improvement to Break Sentinel Control EMT

A minor improvement has been made to the TSX-Plus EMT that connects a completion routine to a break sentinel character (EMT 375, function code 133). The EMT now returns in R0 the address of the completion routine that was previously connected to the break sentinel character. R0 will contain zero if no break sentinel character had been previously selected.

13. DTSUB Improvement

The DTSUB library of DIBOL compatible subroutine calls now handles record numbers greater than 65535.

14. RT-11 Version Dependencies

TSX-Plus may no longer be generated or started from RT-11 Version 3B or any earlier version of RT-11. In addition, device handler SLP files are no longer provided for RT-11 V04, and VM is unsupported under RT-11 V04. If it is absolutely necessary to rebuild a device handler which requires a patch for TSX-Plus, then it is necessary to start from an RT-11 version 5.00 or 5.01 handler source file. The following table indicates the correspondence between SLP file names and RT-11 versions.

RT-11 Version	SLP File Name
5.00	ddd500.SLP
5.01	ddd501.SLP
5.01B	ddd51B.SLP
5.01C	ddd51C.SLP

15. Short-cut Installation Change

The short-cut installation method no longer automatically builds or installs the SYSMON program. In order to use SYSMON, you should perform the TSX-Plus system generation process.

Corrected Problems

16. The following corrections are included in TSX-Plus version 5.1C:

16.1 A problem was corrected which caused some input fields to be prematurely activated. The problem only occurred when field-width activation was declared after characters had been typed ahead and the width-activated field was terminated by an activation condition other than field-width reached. The subsequent field was then activated at the width declared for the previous field.

16.2 A problem was corrected which could cause a fatal system error with the message "SOF-Stack overflow" with an argument value of 2 when using a parallel printer.

16.3 A problem was corrected in the EMT to terminate a performance analysis. If the user-specified histogram buffer was too small to hold all the cells for which the performance analysis was initialized, then a fatal system error would occur with a "SOF-Stack overflow" with an argument value of 3.

16.4 The FORM command accepted a lower case name which could not be mounted using the SPOOL command. The form name is now internally converted to upper case.

16.5 A problem existed when using extended channels above decimal channel number 15, causing these channels to remain active if the program did not properly close them. This happened most frequently in DIBOL or DEC's IB (IEEE-488) software which used these extended channels. Extended channels are now purged on program termination.

16.6 The SET command has been altered to require a minimum of three letters for system SET commands. This prevents conflicts with SET commands for device handlers which require only two characters. A second change to the SET command corrects set options in handlers which read and wrote to their image. These handlers included DU, DX, and DY.

16.7 A rare timing-dependent problem existed when disassociating shared run-time systems while mapped to the I/O page. If an interrupt occurred during the disassociation of shared run-times, the real-time interrupt routine could have been entered without the proper mapping to the I/O page. Because of the problem, one of two errors would occur when attempting to access registers in the I/O page. If the interrupt service routine was a FORK process, a fatal system error would occur with the error message "RIT-Trap in real-time interrupt service routine". If the interrupt service routine was a completion process, the user's job would exit with a "?MON-F-Trap to 4". This problem has been corrected.

16.8 An error was discovered in the prioritization of device handler time out code. The error could cause suspension of device handler FORK processing to execute a device handler time out which was not synchronized to any user (job number of zero). The time out priority has been altered so that the device time out will be queued at the same priority as normal device FORK code (priority of decimal 50) and therefore be prevented from suspending device handler FORK code in process.

16.9 The BACKUP command was corrected to disallow the entry of invalid switches following the second device name. Both the LINK/LIBRARY:file and EXECUTE/LINKLIBRARY:file commands did not select the correct output file name. This has been corrected. The MAKE, MUNG, and TECO commands no longer prompt for a file name if the file specification is omitted. This is compatible with the RT-11 implementation. The CCL expansion of the LINK/PROMPT command has been corrected so that the switch is no longer positionally dependent.

16.10 Condition codes are now preserved across a .INTEN call.

16.11 The method used to return keyboard command error status to IND in the <EXSTAT> symbol has been changed. This should allow more complete error checking by IND than was previously available.

16.12 A problem has been corrected which caused the IND database to become corrupted when more than one user was simultaneously using IND. This problem occurred only with the version of IND that was distributed with RT-11 version 5.1C; the 5.1B version worked correctly.

16.13 TSAUTH has been changed to disallow blanks within user names.

16.14 Keyboard input is no longer written to terminal log files while echoing is suppressed.

16.15 A problem has been corrected in the CL handler which would produce a fatal system error if a .SPFUN was executed which fetched or returned a one-word value, and the specified buffer address was odd. The use of an odd buffer address in this case now causes the handler to set the I/O error flag (just as if a hardware error occurred) and ignore the requested operation. It is permissible to specify an odd buffer address with .READ, .WRITE, and those .SPFUN operations that use the buffer address as the pointer to a byte buffer (e.g., .SPFUN 203).

16.16 A problem has been fixed which could cause gaps to be left in the low-memory job area when the TSX-Plus EMT to lock a job in low memory was used. The problem occurred when a job requested to lock itself in low memory and another job, which was not locked in memory, was currently performing I/O and was located in the low memory area where the locking job needed to be positioned.

16.17 A problem has been corrected which could result in invalid data being placed in the data cache. The problem would occur if a read operation was being performed to bring data into the cache and the operation was aborted by calling the handler abort entry point. Although pending requests for the handler that had not started were flagged as having an I/O error, the current I/O request being processed by the handler was not flagged as being in error. Note: handlers that perform internal queueing must set the error flag in the Channel Status Word if, for any reason, they abort or fail to complete pending I/O requests.

16.18 A problem has been corrected in the CL (Communication Line) handler which could cause it to lose an I/O request if an I/O request was queued to it during a brief critical period while it was beginning processing of an earlier I/O request. The problem was most commonly observed when there were two or more CL units simultaneously active of which at least one was spooled.

16.19 In certain system configurations with particular terminal characteristics (a Q-Bus machine with DHV-11 multiplexer and a VT-10x in smooth scroll mode) it was possible for the system to hang when a job logged off. Changes were implemented to prevent this condition.

16.20 A correction has been made to the MOUNT command to disallow a null file name when mounting a logical disk. For example, the following command is no longer accepted.

```
MOUNT LD DL1:.DSK
```

16.21 A change has been made to the CL (Communication Line) .SPFUN 204 (get handler status). Bit 2 (mask value 4) of the returned status is now 1 if carrier is detected and 0 if carrier is not detected. Previously this bit was always set to 1 regardless of the carrier detect status. This change causes VTCOM to generate "Connection established" and "Connection lost" messages when carrier is detected or lost.

16.22 A problem has been corrected which could cause a fatal system error if a Floating Point Unit (FPU) exception interrupt occurred while the system was processing some other interrupt. This could only happen on systems with floating point units which executed asynchronously from the CPU (such as 11/60's).

16.23 Interpretation of the .CSISPC request has been changed for compatibility with RT-11. It now permits an output specification to include a device name without a file name. Previously, attempts to use an empty file name in an output specification on a file-structured device was flagged as an error. The .CSIGEN request still disallows blank file names in output file specifications on file-structured devices.

16.24 The SQUEEZE and INITIALIZE commands now check to see if any user has a log file open to the device.

16.25 A problem was corrected which previously caused an error when releasing an I/O request which scheduled a user completion routine from a device handler time-out section.

16.26 A problem has been corrected which could result in an incorrect entry in the directory cache if a program such as DUP directly altered the directory on a device. This problem could be produced by use of the CREATE command with the /START or /EXTENSION qualifiers. All directory cache entries for a device are now cleared if a write is performed to the directory blocks on the device.

16.27 A correction has been made to the operation of the .QSET EMT. On return from the EMT, R0 now contains the address of the start of the specified queue area rather than the number of queue elements. Note that the .QSET EMT still has no operational effect on TSX-Plus.

16.28 A problem has been corrected which prevented the creation of PLAS memory regions larger than 2Mb.

16.29 A problem has been corrected which caused .CSISPC or .CSIGEN EMT's to return incorrect information on the stack if an invalid command string was entered.

16.30 A problem has been corrected in the TSX-Plus EMT which sends a message to a terminal (EMT 375, function code 127). The problem prevented a message from being sent to a line that was running KMON when SET TT GAG was in effect for the line.

16.31 A problem has been corrected which caused bit 13 (mask 20000) in the system configuration word 2 (RMON offset 370) to be set whenever TSX-Plus was running on a Q-bus system with at least 256Kb of memory. This bit is now only set when running on a Professional 350/380 computer. One of the symptoms of this problem was that VTCOM used to try to use the XC handler on Q-bus systems with at least 256Kb where it should have only tried to use XC when running on the Professional.

16.32 A problem was corrected which caused the VM handler to size incorrectly if two megabytes or more was reserved for VM use.

16.33 A problem was corrected in the MS handler (TS.MAC) abort entry code which caused the tape handler to lose track of tape position when other jobs would exit.

16.34 SYSMON was altered to display the correct time when using a 50 cycle clock. Selection 10 (CL device display) no longer truncates the user's name.

16.35 The SHOW TERMINAL screen format was altered to prevent terminal wrap from occurring when displaying a line which was not installed.

INDEX

TSX-Plus Version 5.1C Release Notes Addendum, 16 April, 1985

1. MS and MM handlers are now mappable

The device handlers MS and MM may now be loaded into high memory rather than into the low 40Kb of memory in most cases. The exceptions are extended Unibus machines and in combination with the MAPIO option on Q-Bus machines, as is true for all handlers with an internal buffer.

2. $PAGE option recommended

Most terminals use automatic XON/XOFF flow control (including all DEC terminals). In TSX-Plus version 5.1C, system XON/XOFF processing is no longer the default for LA36, ADM3A, Hazeltine or unknown terminal types. System XON/XOFF processing may also be disabled with the SET TT NOPAGE command. For terminals which use XON/XOFF flow control, we recommend including the $PAGE option in the FLAGS macro within the line definition blocks (LINDEF) for those terminals.

3. Use of MAPIO with the DM device handler

The DM device handler .SPFUN function codes 376 and 377 attempt to return a status code into the first word of a user buffer which is one word longer than the actual transfer size. This is incompatible with system I/O mapping. It appears that the only system utility program which issues these functions is DUP (SQUEEZE and INITIALIZE commands). If it is necessary to use the MAPIO option with the DM device handler, we recommend that you only issue INITIALIZE or SQUEEZE commands for DM units from RT-11. Alternatively, you may include the following two lines in the SRFPRG table in TSGEN:

```
.RAD50  /DUP   /
.WORD   AF$MEM
```

4. PRO/TSX-Plus does not support double resolution on the Pro 380

The PI device handler included with PRO/TSX-Plus does not support double resolution graphics on the Professional 380.

5. Additional corrections

5.1 On SYSMON screen 10, CL device display, the CL SET option CTRL is no longer spelled CRTL.

5.2 The DM device handler includes a correction to prevent system crashes when using the IND .TESTDEVICE statement or the FORTRAN FIND statement. These statements issue I/O requests with a word count of 0 (SEEK operation), which caused the handler to clear a random address pointed to by an uninitialized register.

5.3 The nesting of completion routines, described in the 5.1C Release Notes, corrects a problem which prevented SYNCH requests from returning the correct status when I/O was issued from a completion routine. This corrects several symptoms observed with the magtape handlers which used this method.

TSX-Plus Version 5.1

Release Notes Addendum

This addendum describes additional changes made to TSX-Plus version 5.1 subsequent to the printing of the release notes.

October, 1984

TSX-Plus Version 5.1 Release Notes Addendum

1. The following LINK warning message is printed when the TSKMON module is linked:

   ?LINK-W-Multiple definition of $OVRH

   This warning message is expected and does not indicate that any error has occurred.

2. The real-time support facility has been divided into two parts. The primary part which includes all real-time facilities <u>except</u> connection to a real-time interrupt has been made a standard part of the system and is included regardless of the setting of the TSGEN parameter RTVECT. This portion of the real-time support includes the facilities to access the I/O page or map it into a program's virtual address space, to lock a program in memory, to convert a virtual address into a physical address, and the .SPND, .RSUM, and .DEVICE EMT's.

   The second part of the real-time facility provides support for connecting real-time interrupts to interrupt service routines and interrupt completion routines. This portion of the real-time facility is only included in the generated system if the TSGEN parameter RTVECT is set to a non-zero value. This portion of the real-time support system adds approximately 400 bytes of code to the low-memory portion of TSX-Plus.

   In previous versions of TSX-Plus, it was necessary to set the TSGEN parameter RTVECT to a non-zero value in order to use the SYSMON system monitoring and display program. This is no longer the case. All of the facilities needed to run SYSMON are now part of the standard system. RTVECT should only be set to a non-zero value if TSX-Plus jobs need to connect to real-time interrupts.

3. TSX-Plus version 5.1 no longer requires that device handlers be installed in RT-11 in order for them to be loaded and used by TSX-Plus (except for the device handlers for the disk on which the swap and spool files are placed). During TSX-Plus system initialization a test is made to see if the hardware device required by each device handler is present on the system. If the device is present the handler is loaded; if the device is not present, the handler is not loaded. The test for the presence of the hardware device consists of two parts: 1) if the address of the device CSR register is specified in location 176 of block 0 of the device handler, TSX-Plus attempts to access the CSR register and assumes the device is not installed if a trap occurs; 2) if the device handler has an installation code section (starting at location 200 in block 0), the installation code is executed and the handler installation is rejected if the installation code returns with the carry flag set.

4. The following additional corrections were made:

4.1. A problem has been corrected in which the .ELRG EMT stored into and corrupted a cell in the user's virtual address space above 48K bytes. If the user's region was allocated less than 48K bytes, this resulted in a "?MON-F-Kernel mode trap".

4.2. The RT-11 EMT 376 which allows the display of system level error messages has been implemented in TSX-Plus. This corrects the error message generated from the overlay handler to display "?MON-F-Overlay read error".

4.3. A correction has been made to the EMT which aborts detached jobs. Previous versions would allow this EMT to abort jobs which were not detached.

4.4. The implementation of .PURGE has been altered to be compatible with RT-11 when I/O abort is selected (IOABT = 1 or SET IO ABORT). A result of this implementation is that device handlers for all special directory structured devices (such as magtape) will not be entered when a .PURGE is executed. This includes .PURGE requests executed from user programs and those executed from the system as a result of aborting programs.

4.5. A problem has been corrected which caused detached jobs to remain in TI state in KMON when the single line editor was enabled.

4.6. The SETSIZ utility has been modified so that it will no longer abort if a specified file is not found.

4.7. The size of two system stacks has been increased. This reduces the possibility of the fatal error "SOF-Stack overflow".

TSX-Plus Version 5.1

Release Notes

These release notes describe the differences between TSX-Plus version 5.1 and version 5.0. The TSX-Plus Reference Manual and TSX-Plus System Manager's Guide incorporate the information described in earlier release notes. In case of any differences between the release notes and the manuals, the release notes take precedence.

October, 1984

CONTENTS

TSX-Plus Version 5.1 Release Notes

1. A version of TSX-Plus 5.1 (PRO/TSX-Plus) is now available for use with the DEC Professional-350 personal computer. This version permits use of the console terminal, the printer port and the communications port as time-sharing lines. It also includes the new CL handler which may be used to redefine the printer port and communication lines as I/O devices, thus providing maximum flexibility in utilization of system resources. It is distributed as a pregenerated system which includes: three time-sharing lines; three virtual lines, two detached lines and three CL lines (one spooled); generalized data cache; support for 64Kb virtual job regions; virtual array and virtual overlay support (PLAS); shared file record locking; message channels; user defined commands; logical subset disks; single line editor; and the SYSMON utility.

   Supplemental release notes are provided with PRO/TSX-Plus.

2. A Communications Line (CL) handler is included with version 5.1. The CL handler allows Input/Output operations to be performed to serial communication lines connected to DL11, DLV11, DZ11, DZV11, DH11, and DHV11 communication controllers. With the CL handler it is possible to have some lines on a multiplexer used as TSX-Plus time-sharing lines, and other lines on the same multiplexer used to drive I/O devices such as printers, plotters, and modems. It is also possible to use a line as a time-sharing line some of the time and as a communications line at other times. Up to 8 communication lines can be controlled by the CL handler. Some of the important features of the CL handler are summarized below:

   1. Up to 8 communication lines may be controlled through the CL handler. The device names are "CL0:", "CL1:", ... "CL7:". The lines may be connected to any type of communication controller that is supported by TSX-Plus and may share the same multiplexer controllers as TSX-Plus time-sharing lines.

   2. Lines may be dedicated as communication lines or may be switched between use as time-sharing lines and communication lines.

   3. Internal queueing is used within the handler to allow concurrent Input/Output operations to be performed on all of the lines.

   4. The CL handler allows both input (read) and output (write) operations. Full duplex (simultaneous) read and write operations may take place on each line.

   5. The communication lines may be used with the TSX-Plus spooling system to allow spooled output to devices on communication lines.

   6. The CL handler responds to XON/XOFF (ctrl-Q/ctrl-S) control characters to stop and start its transmission and will generate XON/XOFF characters to control the speed of a device transmitting to a CL line.

7. A "binary mode" is available for CL lines to allow full 8-bit, transparent I/O to devices.

8. Modem control is supported. Ring and carrier detect signals may be monitored and data terminal ready (DTR) can be controlled by a program or SET command.

9. The CL handler is implemented as a system virtual overlay, minimizing the amount of code and data that is required in the unmapped portion of the system. Typically the size of the unmapped portion of the CL handler is 200 to 500 bytes (depending on the number of lines supported) and the mapped portion is approximately 2200 bytes.

10. The CL handler can be used as a replacement for the LS and XL handlers (the XL handler is used with the RT-11 VTCOM program).

System generation considerations. The number of available CL units is established at the time a TSX-Plus system is generated. CL units may be connected with dedicated CL lines when the system is generated or may be left "unattached" in which case a SET command or system service call can be used to cause them to "take over" free time-sharing or CL lines after the system is started. The number of unattached CL units is declared by the value specified for the CLXTRA parameter in TSGEN. The number of CL units connected to dedicated CL lines is declared by the fourth parameter of the TBLDEF macro. The total number of CL units (<fourth TBLDEF parameter> + <CLXTRA>) may not exceed 8. The CL unit numbers range from 0 up to (total CL units - 1).

The TBLDEF macro has been changed to accept an additional parameter which specifies the number of dedicated CL lines. The four parameters are: 1) number of physical time-sharing lines; 2) number of virtual time-sharing lines; 3) number of detached jobs; 4) number of dedicated communication lines.

Dedicated communication lines are reserved for use only as CL lines; they may not be used as time-sharing lines. However, the CL units which are initially associated with dedicated CL lines may be reassigned to other lines. Dedicated communication lines are declared in TSGEN in much the same fashion as time-sharing lines. The difference is that the CLDEF macro is used to declare dedicated communication lines whereas the LINDEF macro is used to declare time-sharing lines. The form of the CLDEF macro is slightly different depending on whether the line is connected to a DL(V)11 type communications controller or to a multiplexer. The two forms of the CLDEF macro are shown below:

DL(V)11:

```
        CLDEF     unit_number,vector_address,receiver_address
```

Multiplexer:

```
        CLDEF     unit_number,mux_line
```

The "unit_number" parameter specifies the CL I/O device unit number by which this line will be referenced. For example, unit number 3 is referred to by the name "CL3:". The units may be specified in any order but the unit numbers must be less than the total number of communication lines (<fourth TBLDEF parameter> + <CLXTRA>). That is, if 3 communication lines are specified, they must have unit numbers 0, 1, and 2.

For DL(V)11 lines, "vector_address" is the address of the receiver input vector and "receiver_address" is the address of the receiver control and status register (CSR). These two parameters are the same as would be specified with a LINDEF macro for a DL(V)11 line. For multiplexer lines, the "mux_line" parameter specifies the number of the port within the multiplexer. This corresponds to the single parameter that would be specified with a LINDEF for a multiplexer line.

A CLDEF macro is used to start the definition of a communications line. The CLEND macro is used to terminate the communications line definition block just as a LINEND macro is used to terminate a time-sharing line definition started with a LINDEF macro. The SILO, BUFSIZ, SPEED, NAME, and FLAGS macros may be used between the CLDEF and CLEND macros to specify parameters for the line.

The BUFSIZ macro specifies the sizes of the output character ring buffer for the line. If no BUFSIZ macro is used, the values set for the CLORSZ parameter is used by default. The recommended size of the output ring buffer is ((3*baud_rate)/1000+2). The form of the BUFSIZ macro for a dedicated CL line is:

```
    BUFSIZ    0,output_buffer_size
```

The SILO macro may be used to specify the size of the input character silo for a CL line. It has the same form and options as for a normal time-sharing line. If a CL line is only used for output (such as to drive a printer) the input silo size may be set to 0 (zero). (A CL line with an input silo size of 0 can still respond to XON/XOFF from an external device.)

The SPEED macro has the same form and options as used to provide parameters for a time-sharing line. It should be used to specify the transmit/receive baud rate for multiplexer lines.

The FLAGS macro can be used to initialize some options for communication lines. It has the same form as is used with time-sharing lines, but only the $FORM and $TAB flags are operational. These flags correspond to the SET CLn FORM and SET CLn TAB keyboard command settings.

For example, the following section illustrates a possible definition for a system with two time-sharing lines and two communication lines.

```
TBLDEF  2.,1.,1.,2.        ;#real, #virtual, #detached, #comm

LINDEF  60,177560,OPER     ;Operator's console is 1st T/S line
NAME    <Console>          ;Descriptive name for line
FLAGS   $PRIV!$START       ;Auto startup and operator privilege
CMDFIL  START1.TSX         ;Startup command file
LINEND

CLDEF   1,340,176540       ;Define CL1 on a DL(V)11
NAME    <LS120 printer>    ;Descriptive name for line
CLEND

DZDEF   300,160040         ;Begin DZ11 multiplexer lines

LINDEF  0                  ;Second time-sharing line, mux port 0
NAME    <Al Jackson>       ;Descriptive name for line
SPEED   S9600              ;9600 baud
LINEND

CLDEF   0,1                ;Define CL0, mux port 1
NAME    <Printer 2>        ;Descriptive name for line
SPEED   S9600              ;9600 baud
FLAGS   $FORM              ;Printer supports form-feeds
SILO    0,0,0              ;Don't need an input silo
CLEND

MUXEND                     ;End of DZ11 lines
```

The CL handler is automatically included in the system if any communication lines are specified in TSGEN. Do not specify CL as a device using the DEVDEF macro in TSGEN. Since the CL handler is a system virtual overlay, there is no CL.SYS or CL.TSX file.

Output may be spooled to a device connected to a communications line by specifying the CL device name (e.g., "CL1") as one of the devices with the SPOOL macro in TSGEN. The device name "CL" is functionally equivalent to "CL0". Attempts to use a CL unit which is not currently associated with a line return an error status just as if the CL device was not recognized by the system.

Once a system has been generated with communication lines, the lines may be accessed as normal devices using the names CL0, CL1, etc. If the CL handler is used to drive the system printer, it is convenient to use an assign command to assign the logical name LP to the corresponding CL device.

If the CL handler is used with the VTCOM program, it is necessary to assign the logical name XL (or XC if on the Professional) to the CL unit controlling the communications line. The NOLFOUT option should be specified for a CL line used with VTCOM. For example, the following commands would be appropriate to direct VTCOM to use line CL0:

```
.SET CL0 NOLFOUT
.ASSIGN CL0 XL
```

I/O operations. The .READ/.READC/.READW and .WRITE/.WRITC/.WRITW EMT's may be used to perform standard read/write operations to the CL lines. The CL handler allows full duplex input/output operation, which means that read and write operations may be simultaneously active on a CL line.

When a .READ[C/W] EMT is used to read from a CL line, the operation is complete when the requested number of words have been accepted or a control-Z character is received.

The input character silo is used to store characters received from the line. This buffer prevents characters from being lost during the interval when one read EMT is completed and another is issued to the CL handler.

The CL handler responds to received XON (ctrl-Q) and XOFF (ctrl-S) characters, starting and suspending transmission to synchronize its rate with the device connected to the line.

.SPFUN operations. The following special function codes (.SPFUN EMT's) are recognized by the CL handler. The special functions apply to the specific CL unit to which the channel was opened.

Function 201 -- Clear handler. This function clears the internal handler flag that says an XOFF (ctrl-S) character has been received and transmits an XON (ctrl-Q) to the CL device. The SET CL RESET keyboard command can be used to perform this function.

Function 202 -- Control break transmission. This function starts or stops the transmission of a break signal. The word count specified with the .SPFUN controls whether transmission of a break signal is started or stopped. If the word count is non-zero, break transmission is started; break transmission continues until another .SPFUN is done with function code 202 and a word count of zero.

Function 203 -- Read with byte count. This function performs a read operation but the "word count" value specifies a byte count instead. This function does not complete until at least one byte is read. However, if a byte count greater than one is specified, bytes are moved from the input ring buffer until either the specified byte count is satisfied or the input ring buffer is emptied. If fewer than the requested number of bytes are available, the remainder of the buffer is filled with nulls. The control-Z character does not signal end of file for this type of read -- control-Z is read as an ordinary character.

Function 204 -- Get handler status. A status code is stored into the first word of the buffer specified with this function. The meaning of the flag bits is as specified below:

bit 0: 1 ==> XOFF has been sent to stop transmission.
bit 1: 1 ==> XOFF has been received from the remote device.

Function 205 -- Terminate I/O to the line. This function "turns off" a communication line. The input ring buffer is emptied (its contents are discarded) and a flag is set causing any other characters received from the line to be discarded. Data Terminal Ready (DTR) status is dropped. The line will be turned on again whenever another I/O operation is performed to it.

Functions 250 and 251 -- Control option flags. These special functions are used to set and clear handler option flags. Function 250 sets the specified flag bits, function 251 clears the specified flag bits. The flag bits correspond to handler SET options. If the option flag is cleared (0), this corresponds to the NOoption setting. The bit positions of the options are shown in the following table. The option flags are contained in a one word buffer for the .SPFUN. See the descriptions of the SET commands for detailed descriptions of the option functions.

Bit	Mask	Option	Function summary
0	000001	FORM	Device supports form feeds
1	000002	TAB	Device supports tabs
2	000004	LC	Send lower case characters
3	000010	LFOUT	Transmit line feed characters
4	000020	LFIN	Accept line feed characters
5	000040	FORM0	Send form feed on block 0 write
6	000100	BINOUT	Binary output mode
7	000200	BININ	Binary input mode
8	000400	CR	Send carriage return characters
9	001000	CTRL	Send control characters
10	002000	DTR	Raise Data Terminal Ready
11	004000	EIGHTBIT	Allow 8 bit character transmission

Function 252 -- Set page length. This function performs the operation of the SET CL LENGTH=n command. The .SPFUN must have a one word buffer containing the number of lines per page.

Function 253 -- Set skip lines. This function performs the operation of the SET CL SKIP=n command. The .SPFUN must have a one word buffer containing the number of lines to skip at the bottom of the page.

Function 254 -- Set page width. This function performs the operation of the SET CL WIDTH=n command. The .SPFUN must have a one word buffer containing the line width.

Function 255 -- Get modem status. This function is used to check on the status of a modem connected to a CL line. The modem status is returned into the first word of the buffer specified with the .SPFUN. The flag bits returned are described below:

Bit	Meaning when set
0	Ring indication
1	Carrier is detected
2	Data Terminal Ready is asserted

Function 256 -- Set line speed. This function is used to set the transmit/receive speed for a CL line. This .SPFUN must have a one word buffer containing a speed code. The following baud rates are represented by the indicated speed codes (speed code values are shown in decimal): 75=1, 110=2, 134.5=3, 150=4, 300=5, 600=6, 1200=7, 1800=8, 2000=9, 2400=10, 3600=11, 4800=12, 7200=13, 9600=14, 19200=15. This .SPFUN can only be used for lines connected to hardware controllers that support programmable baud rates such as: DZ11, DZV11, DH11, DHV11, DLV11-E, and the Pro printer and communication ports. A baud rate of 19200 is not supported by DEC DZ(V)11 controllers.

Function 257 -- Abort all pending read and write operations issued by the job executing the .SPFUN on the CL unit.

Function 260 -- Read a line of input. This special function reads a line of input terminated by a carriage return character. The "word count" value specified with this special function is interpreted as a byte count. The read terminates when any of the following conditions is met:

1. A carriage return character is received. The carriage return is stored in the buffer and the remainder of the buffer is null filled.

2. The buffer is filled before a carriage return is received.

3. A control-Z is received.

<u>SET Options.</u> The following options may be used with a SET CLn command. The SET command controls options for a specific CL unit. Sets to CL units are not permanent and must be specified each time the system is started. A detached job start-up command file is a convenient way to perform CL sets each time the system is started. Operator privilege is required to perform a SET to a CL unit.

[NO]BININ -- Sets (BININ) or resets (NOBININ) binary input mode. In binary input mode, 8 bit data characters are passed directly to the user's data buffer. Control characters such as ctrl-S, ctrl-Q, and ctrl-Z are treated as normal data characters. A read operation is only completed when the specified word count is satisfied. The default mode is NOBININ.

[NO]BINOUT -- Sets or resets binary output mode. In binary output mode 8 bit data characters are sent to the CL line without modification. In this mode the CL handler does not send XON and XOFF characters to control the speed of the remote device but will respond to received XON/XOFF characters unless binary input mode is also in effect. The default mode is NOBINOUT.

[NO]CR -- NOCR mode causes the CL handler to discard carriage return characters in data being written to a CL device. Carriage return characters received from a CL line are not affected. The default mode is CR.

[NO]CTRL -- NOCTRL mode causes the CL handler to discard control characters in data being written to a CL device. It does not affect characters received from a CL line. Paper and head movement control characters (BS, TAB, LF, FF, and CR) are not affected by the setting of CTRL. The default mode is CTRL.

[NO]DTR -- The DTR option causes the CL handler to raise the Data Terminal Ready signal for the modem connected to the specified CL line. NODTR causes the Data Terminal Ready signal to be dropped. The DTR signal is automatically asserted if a read or write operation is directed to a CL line even if that line has been set NODTR. The default (initial) mode is NODTR.

[NO]EIGHTBIT -- If the EIGHTBIT option is selected, all eight bits of received characters are passed to programs. If NOEIGHTBIT is specified, characters are masked to seven bits. If EIGHTBIT mode is selected, the device connected to the CL line should be set for 8 bit characters and no parity support.

[NO]FORM -- The FORM option specifies that the CL device can handle form feed characters. If NOFORM is specified, form feed characters are translated into enough line feed characters to cause the line position to advance to the top of the next page as specified by the LENGTH=n option. The default mode is controlled by the $FORM flag used with the FLAGS macro in TSGEN. CLXTRA lines default to NOFORM.

[NO]FORM0 -- If the FORM0 option is in effect, the CL handler generates a form feed each time a write operation specifies a block number of 0. The default mode is NOFORM0.

[NO]LC -- The LC option allows lower case characters to be transmitted to the CL line. NOLC causes lower case characters to be translated to upper case before transmission. Lower case input characters are not converted. The default mode is LC.

LENGTH=n -- Specifies the number of lines on a page. This option is used with the NOFORM option to cause the CL handler to use line feed characters to simulate a form feed. If LENGTH=0 and NOFORM are specified, no form feed simulation takes place and form feed characters are discarded. The

LENGTH option also is used when the SKIP=n option is used. The default page length is 66 lines.

[NO]LFIN -- The NOLFIN mode causes the CL handler to discard line feed characters received from the CL device. The default mode is LFIN.

[NO]LFOUT -- The NOLFOUT mode causes the CL handler to discard line feed characters that are written to CL devices if the line feed character immediately follows a carriage return. The default mode is LFOUT. The NOLFOUT mode should be used when using a CL line with VTCOM.

LINE=n -- The LINE option specifies the number of the TSX-Plus terminal line to which the CL unit is to be connected. The parameter n must be the number of a physical time-sharing line or a dedicated CL line. You may not assign a CL unit to a line which has a logged on time-sharing user or which has some other CL unit assigned to it. If "LINE=0" is specified, the CL unit is disassociated from any particular line and is left unassigned. Time-sharing lines may be temporarily used as CL units and then restored to their normal status by disassociating the CL unit from the line. The SHOW CL command can be used to display which CL units are connected to which lines.

RESET -- The RESET option performs the same functions as special function (.SPFUN) number 201. It resets the flag that says a control-S (XOFF) character has been received and transmits a control-Q (XON) character.

SKIP=n -- The SKIP option specifies the number of lines to skip at the bottom of each page. The page length is specified by the LENGTH=n option (or defaults to 66). The default mode is SKIP=0 which causes no lines to be skipped at the bottom of each page.

SPEED=n -- The SPEED option specifies the transmit/receive speed for the CL line. The available baud rates are: 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200. This option can only be used for lines connected to hardware controllers that support programmable baud rates such as DZ11, DZV11, DH11, DHV11, DLV11-E, and the Pro printer and communication ports. A baud rate of 19200 is not supported by DEC DZ(V)11 controllers.

[NO]TAB -- The TAB option specifies that the CL device can respond to tab characters. If NOTAB is specified, tab characters are translated into enough space characters to position the output column to the next multiple of 8. The default mode is controlled by the $TAB flag used with the FLAGS macro in TSGEN. CLXTRA lines default to NOTAB.

TOP -- The TOP option is used to declare that the paper is positioned at the top of the page. It is necessary to do this when the paper is repositioned and the device does not support hardware form feeds.

WIDTH=n -- Specifies the maximum line width. Output characters are discarded if they would print beyond the specified line width. The default is WIDTH=0 which allows unlimited width lines.

Redirecting CL and time-sharing lines:
A system service call (EMT) is available to allow a program to assign a CL unit to a particular line. The form of the EMT is:

```
EMT     375
```

with R0 pointing to an argument block of the following form:

```
.BYTE   0,155
.WORD   CL_unit
.WORD   line_number
```

where "CL_unit" is the CL unit number, and "line_number" is the number of a TSX-Plus time-sharing line or dedicated CL line. If the specified line number is 0 (zero), the CL unit is disassociated from any line. Operator privilege is required to use this EMT.

If an error is detected, the C-flag is set on return and the following error codes are returned:

Code	Meaning
1	User issuing the EMT does not have operator privilege
2	An invalid CL unit number was specified
3	An invalid line number was specified
4	The specified line is already assigned to a CL unit
5	A time-sharing user is logged onto the specified line
6	The specified CL unit is currently busy

CL units specified using the CLDEF macro in TSGEN are initially connected to dedicated CL lines. Note that although the lines are dedicated to use by CL, the CL units which are initially assigned to these lines may be reassigned to other lines. The unallocated CL units declared by use of the CLXTRA parameter in TSGEN are initially not associated with any line. The SET CLn LINE=n keyboard command or the system service call (EMT) can be used to assign any CL unit to any free time-sharing line or free dedicated CL line. Thus it is possible to use a line as a TSX-Plus time-sharing line during certain portions of the day and then assign a CL unit to the line and use it to drive a modem or other device during other portions of the day. Dedicated CL lines use less space than time-sharing lines but may only be accessed as CL units.

The following example commands illustrate how CL unit 1 can be assigned to time-sharing line 2. The logical name "LP" is then assigned to CL1 so that the PRINT command will direct output through CL1. CL1 can be declared to be a spooled device in TSGEN:

```
.SET CL1 LINE=2
.ASSIGN CL1 LP
```

The SHOW CL and SHOW TERMINALS keyboard commands can be used to display information about which CL units are associated with which lines. The SHOW CL command also indicates if a CL unit is spooled and lists the options which are set for the unit. An example of the SHOW CL command is shown below:

```
.SHOW CL

Unit  Line  Job          Options
----  ----  ---  ---------------------------------
 CL0   3     1   [TAB,LC,LFIN,CR,CTRL]
 CL1  none
 CL2   2     6   (spooled) [LC,LFOUT,LFIN,CR,CTRL]
```

The "Line" column shows which line the CL unit is assigned to. The "Job" column provides the number of a job which has either allocated the CL unit or is currently using the CL unit.

3. Support is provided for DH11 multiplexers. The TSGEN specification for a DH11 multiplexer is very similar to that used to define a DZ11 multiplexer, except a DHDEF macro is used to begin the group of lines rather than a MUXDEF macro. The DHDEF macro requires four arguments:

   1. Address of interrupt vector for DH11.
   2. Address of CSR register for DH11.
   3. Address of interrupt vector for associated DM11.
   4. Address of CSR register for DM11.

   A DM11 is the modem control section of a DH11 multiplexer. With some DH11's, the DM11 is optional. If your DH11 does not have an associated DM11, specify 0 (zero) for the third and fourth arguments of the DHDEF macro.

   For example, the following TSGEN declarations would define a DH11 multiplexer with an interrupt vector address of 330 and device status registers starting at 160060, and an associated DM11 with an interrupt vector of 310 and status register of 170520:

```
DHDEF   330,160060,310,170520  ;Begin DH11 mux lines

LINDEF  0                      ;Mux line 0
NAME    <Work room>            ;Descriptive name for line
SPEED   S9600                  ;9600 baud
FLAGS   NRMFLG                 ;Normal terminal flags
CMDFIL  LINE0.TSX              ;Start-up command file
TRMTYP  VT100                  ;Terminal type
LINEND

LINDEF  1                      ;Mux line 1
NAME    <Dial up line>         ;Descriptive name for line
FLAGS   NRMFLG!$AUTO           ;Autobaud
CMDFIL  LINE1.TSX              ;Start-up command file
TRMTYP  VT100                  ;Terminal type
LINEND

MUXEND                         ;End of lines on DH11 mux
```

For consistency, a "DZDEF" macro has been defined which is equivalent to MUXDEF. The DZDEF macro is the preferred way to begin the line definitions for a DZ11 multiplexer.

Note that the system interface to a DH11 multiplexer is different from that for a DHV11 multiplexer (they have a different control register structure). Thus a DHV11 is not the same as a DH11 installed on a Q-bus system. In fact, there are DH11 compatible multiplexers available for use on Q-bus systems. Therefore, choose the DHDEF or DHVDEF macros based on the type of multiplexer, not on the type of system on which they are installed.

TSX-Plus uses the DMA output capabilities of the DH11 multiplexer which reduces system overhead for character output by about 14% compared with a DZ11 driving a time-sharing line or about 20% for a CL line.

4. Support is now provided for DHV11 multiplexers. The TSGEN specification for a DHV11 multiplexer is very similar to that used to define a DZ11 multiplexer except a DHVDEF macro is used to begin the group of lines rather than a MUXDEF macro. For example, the following TSGEN commands would define a DHV11 multiplexer with an interrupt vector address of 350 and device status registers starting at 160020:

```
DHVDEF  350,160020        ;Begin DHV11 mux lines

LINDEF  0                 ;Mux line 0
NAME    <Work room>       ;Descriptive name for line
SPEED   S9600             ;9600 baud
FLAGS   NRMFLG            ;Normal terminal flags
CMDFIL  LINE0.TSX         ;Start-up command file
TRMTYP  VT100             ;Terminal type
LINEND

LINDEF  1                 ;Mux line 1
NAME    <Dial up line>    ;Descriptive name for line
FLAGS   NRMFLG!$AUTO      ;Autobaud
CMDFIL  LINE1.TSX         ;Start-up command file
TRMTYP  VT100             ;Terminal type
LINEND

MUXEND                    ;End of lines on DHV11 mux
```

For consistency, a "DZDEF" macro has been defined which is equivalent to MUXDEF. The DZDEF macro is the preferred way to begin the line definitions for a DZ11 multiplexer.

Note that the system interface to a DH11 multiplexer is different from that for a DHV11 multiplexer (they have a different control register structure). Thus a DHV11 is not the same as a DH11 installed on a Q-bus system. In fact, there are DH11 compatible multiplexers available for use on Q-bus systems. Therefore, choose the DHDEF or DHVDEF macros based on the type of multiplexer, not on the type of system on which they are installed.

TSX-Plus uses the DMA output capabilities of the DHV11 multiplexer which reduces system overhead for character output by about 14% compared with a DZ11 driving a time-sharing line or about 20% for a CL line.

5. Support is now provided for eight bit characters to be transmitted to and received from terminals. A line may be enable to send and receive 8 bit characters by use of a sysgen flag or a keyboard command. The $8BIT flag may be used with the FLAGS macro within a line definition block in TSGEN to enable 8 bit character I/O. Also, the following keyboard command can be used to enable 8 bit I/O for a line:

   SET TT EIGHTBIT

   The following keyboard command sets a line for 7 bit I/O:

   SET TT NOEIGHTBIT

   When in 8 bit mode, all 8 bits of each character are passed to and from the terminal. The null character (000) may not be received from a

terminal, irrespective of 7 or 8 bit mode. If VTxxx escape sequences are declared to be a TSX-Plus activation condition, the character 377 (octal) also may not be received. Note that control codes such as control-C, control-S, control-Q, delete, etc. are still processed by the system.

When using 8 bit mode, terminals should be set for 8 bit characters without parity. VT200 terminals should be set to VT200 mode with 7 bit control characters.

6. "Autobaud" automatic speed selection is now supported for time-sharing lines connected to hardware controllers that have programmable speed selection such as DZ11, DZV11, DH11, DHV11, DLV11E, and the Pro printer and communication ports. When autobaud is specified for a line, the line's speed is set to match the speed of the terminal connected to the line. When the carriage return character is received to start up a line the system examines the (possibly garbled) character and uses the received character value to determine the speed of the transmitting terminal. The line's speed is then set to that value and the line is started.

   To specify automatic speed selection for a line, use the "$AUTO" flag with the FLAGS macro within the line definition block in TSGEN. The SET TERMINAL command described below can also be used to set or cancel automatic speed selection.

   The following baud rates are supported by automatic speed selection: 110, 300, 1200, 2400, 4800, 9600, and 19200. It is necessary to type two consecutive carriage returns to start a line with autobaud if the speed is less than 1200 baud. A speed of 19200 baud is not recommended for machines slower than an 11/73 or 11/44 and this speed is not supported by some hardware controllers such as the DEC DZ11.

7. Several enhancements have been made to the SET TERMINAL (or SET TT) command. The command syntax has been changed to allow specification of a line number to allow the SET TERMINAL command to affect a line other than the one issuing the command. The form of the command used to set an option for another line is:

   SET TERMINAL <u>line</u> <u>option</u>

   where <u>line</u> is the number of the primary line to which the SET command is directed and <u>option</u> is one of the valid options for the SET TERMINAL command. For example, the following command sets the speed for line number 4 to 4800 baud:

   SET TERMINAL 4 SPEED=4800

   Note that a space is required between the word TERMINAL (or TT) and the line number. If no line number is specified, the option applies to the terminal that is issuing the command.

The operation of the SET command is slightly different depending on whether a line number is specified or not. If a line number is specified (even if it corresponds to the terminal issuing the command), the parameter change becomes permanent and will remain in effect between times when one job logs off and another logs on (but not across a system restart). If a line number is not specified, the change is temporary and will be reset when the current job logs off (except for line speed which is always permanently changed). Operator privilege is required to set parameters for a line other than the one issuing the command or to make permanent changes.

The following new options may now be specified with SET TERMINAL:

[NO]AUTOBAUD -- If the AUTOBAUD option is specified for a line, then automatic baud rate selection will be done to set the speed of the line when a line logs on. The NOAUTOBAUD option cancels automatic baud rate detection. Use of the SPEED=n option also cancels automatic baud rate detection. Autobaud may be selected as the default mode for a line by use of the $AUTO flag in the TSGEN line definition.

SPEED=n -- The SPEED option specifies the transmit/receive speed for the line. The available speed values are: 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200. This option can only be used for lines connected to hardware controllers that support programmable baud rates such as DZ11's, DZV11's, DH11's, DHV11's, DLV11E's, and the Pro printer and communication ports. A speed of 19200 baud is not supported by DEC DZ11 controllers. Use of the SPEED option cancels automatic baud rate detection for a line; the AUTOBAUD option may be used to restore automatic baud rate selection.

[NO]DEAD -- If a line is set DEAD no further logons are allowed on the line. The NODEAD option can be used to resurrect a line. A CL unit may not be assigned to a line that is flagged as dead. Operator privilege is required to use this option.

[NO]EIGHTBIT -- The EIGHTBIT option enables all eight bits of received characters to be passed to the program. NOEIGHTBIT mode causes characters to be masked to seven bits.

[NO]PHONE -- The PHONE option specifies that the line is connected to a modem. NOPHONE specifies that the line is hardwired to a terminal. Operator privilege is required to use this option.

[NO]PRIVILEGE -- These options specify whether or not a line is to have operator privilege. Operator privilege is required to issue this option.

8. A system service call (EMT) has been added to allow the transmit/receive speed for a line to be controlled. The form of the EMT is:

```
EMT     375
```

With R0 pointing to an argument block of the following form:

```
.BYTE   0,154
.WORD   line_number
.WORD   speed_code
```

Where "line_number" is the number of the line for which the speed is to be set. If the line number is zero, the speed is set for the line on which the program issuing the EMT is running. Operator privilege is required to change the speed of a line other than the one on which the program is running. "Speed_code" is a value in the range 0 to 15 (decimal) that indicates the desired speed. The following baud rates are represented by the indicated speed codes (speed code values are shown in decimal): 75=1, 110=2, 134.5=3, 150=4, 300=5, 600=6, 1200=7, 1800=8, 2000=9, 2400=10, 3600=11, 4800=12, 7200=13, 9600=14, 19200=15. This EMT is only functional for lines connected to hardware controllers that support programmable baud rates such as DZ11, DZV11, DH11, DHV11, DLV11-E, and the Pro printer and communication ports. A speed of 19200 baud is not supported by DEC DZ(V)11 controllers.

9. TSX-Plus now supports programmable line speed control for lines connected to DLV11E controllers. The SPEED macro may be used within the line definition blocks for these lines to set the initial line speed. The SET TERMINAL and the SET CL commands may be used to change the line speed after the system is started.

10. The following changes have been made to dial-up line support:

    1. A line generated as a dial-up line ($PHONE flag specified for line in TSGEN) may now also be used with a hardwired terminal. If a carriage return is received from the line when the line is not logged on, and carrier-detect is not asserted by the modem, the system assumes a local terminal is connected to the line and does not log off the line if carrier is lost. If carrier-detect is asserted when the line logs on, the system monitors carrier detect and logs the job off if carrier is lost.

    2. A carriage return must now be pressed to log on a dial-up line after the phone is answered.

    3. A new sysgen parameter, OFFTIM, controls how long a user may remain connected to a dial-up line in a logged-off state before the phone is hung up.

These changes make it possible to generate lines with the $PHONE flag and then use them at different times with modems or hardwired terminals.

11. The function of "tape" mode terminal control has been changed. Tape mode is selected by use of the $TAPE sysgen flag or the SET TERMINAL TAPE keyboard command. In previous versions of TSX-Plus, tape mode had three effects:

    1. Enable computer to generate XON and XOFF characters to stop and start transmission to the computer.

    2. Suppress echoing of bell characters as the terminal input buffer became full.

    3. Ignore received line feed characters.

    In TSX-Plus version 5.1, the only effect of tape mode is to cause the system to ignore line feed characters.

    The system now always generates XON and XOFF characters to control the flow of characters to the computer and a bell character is only echoed when the terminal input buffer overflows and an input character is lost.

12. A new program debugging facility is included with TSX-Plus version 5.1. This new facility replaces the old TSODT program. The major features of the new debugging facility are:

    1. The new debugger does not share memory space with the program being debugged (it is a system overlay). This means that the debugger can be used with programs using up to 64Kb of memory. It can also be used with real-time programs that access the I/O page and with programs that use shared run-time systems or PLAS regions.

    2. The debugging facility is invoked by use of the "RUN/DEBUG" keyboard command. It is not necessary to link the debugger with the program being debugged.

    3. The execution of a program can be interrupted at any time by typing control-D. This causes an entry to the debugging system (similar to hitting a breakpoint) where the status of the program can be checked and then execution continued if desired.

    4. When enabled by a keyboard command (SET CTRLD DEBUG), the debugger may be invoked by typing control-D even during execution of a program which was not started under control of the debugger.

5. A data "watchpoint" facility is included which allows a data item to be monitored and a break to occur when the value of the data item changes.

6. Instructions can be decoded into symbolic form.

7. Program traps to 4 and 10 are caught by the debugger rather than causing a program abort and entry to the keyboard monitor.

The commands and syntax of the debugging facility are similar to those used by ODT. However, not all ODT commands are implemented and some additional features are provided.

A program is started with the debugger by typing

```
RUN/DEBUG program
```

or

```
R/DEBUG program
```

When the program is started, the following prompt is printed at the terminal:

```
TSX-Plus debugger

DBG:
```

The "DBG:" prompt indicates the debugger is in control and is waiting for a command. On entry to the debugger, the program start address is in R0 and is set so that the ";G" command can be used to initiate execution without having to specify a starting address.

At any time during the execution of a program with the debugger, the execution of the program may be interrupted by typing control-D. This causes an entry to the debugger similar to a breakpoint. The status of the program can be checked and its execution resumed by typing ";P". If a system service call (EMT) is being executed when control-D is typed, the service call runs to completion and the debugger is entered as the system is about to return from the EMT to the program.
Normally, control-D only causes a breakpoint if the executing program was started with the debugger. However, it is possible to enable control-D to cause a debugger breakpoint for any program, even if the program was not started with the debugger. To enable control-D to interrupt the execution of any program run from your time-sharing line, issue the following keyboard command:

```
SET CTRLD DEBUG
```

To disable this function, use the following command:

```
SET CTRLD NODEBUG
```

The following commands are available and have a syntax and function similar to those used by ODT:

address/	Display the contents of the addressed word.
address\	Display the contents of the addressed byte.
value<return>	Store the value into the currently open cell. No value specified closes the current location without changing its contents.
value<line feed>	Store the value into the currently open cell and then advance to the next word or byte. No value specified advances to the next location without changing the value of the current location.
value^	Store the value into the currently open cell and then advance to the previous word or byte. No value specified backs up to the previous location without changing the contents of the current location.
Backspace	Equivalent to up-arrow ("^").
@	Open the cell whose address is the contents of the currently open cell.
_ (underscore)	Open the cell whose address is specified by the contents of the currently open cell plus the current location plus 2.
address[	Open the cell whose address is specified and display its contents as a symbolic instruction. If no address is specified, the contents of the currently open cell are decoded into symbolic instruction form.
address]	Open the cell whose address is specified and display its contents as a symbolic instruction. If no address is specified and a cell is currently open as an instruction, close the current cell and open the next cell and display its contents in symbolic instruction format. The position of the next cell opened is determined by the number of bytes used by the instruction in the currently open cell.
valueTAB	The TAB key causes the next instruction to be executed in single step mode. It is not necessary to issue the ";1S" command to single step using the TAB key. If a non-zero value is specified as a value before TAB is typed, the value is used to calculate the address of the next instruction to be single stepped (similar to ;1S, then value;P). If 0 (zero) is specified as a

value before TAB is typed, and the next instruction is a JSR (CALL), then a temporary breakpoint is set beyond the JSR to catch the return from the called subroutine and the subroutine is executed without single stepping. This can be used to avoid single stepping through subroutines that are called by code that is being single stepped.

address;nR — Set relocation base register n to address. The relocation registers are numbered 0 through 7.

nR — Convert the contents of the currently open cell into an offset relative to relocation register n.

n! — Convert the current address to an offset relative to relocation register n.

address;nB — Set an instruction breakpoint at address. There are 8 instruction breakpoints numbered 0 through 7. Set a breakpoint to address 0 to clear the breakpoint.

address;G — Begin program execution at the specified address.

repeat;P — Proceed from a breakpoint. If the repeat count is specified, don't break again until repeat breakpoints have been hit. If repeat is not specified, proceed to the next breakpoint.

;nS — Set or reset single stepping mode. If n is 1 (or any other non-zero value), the debugger begins doing single stepping and will step one instruction for each execution of the ";P" command. If n is 0 or is not specified, single stepping mode is reset and the program will continue normal execution when the ";P" command is issued.

X — Convert the currently displayed value from a RAD50 value into a three character ASCII string. Optionally accept a new three character replacement string.

address;valueM — Set a data watchpoint. The word specified by address is monitored. If value is specified, a data watchpoint occurs when the value of the word matches value. If value is not specified, a data watchpoint occurs any time the value of the word changes.

Address values may be specified absolutely or in the form "n,offset" where n is a relocation register number and offset is the offset relative to the relocation register. When an instruction is decoded into symbolic form, relative addresses are displayed in the form "[n,offset]" as an offset from the nearest lower relocation register. The $F register can be used to control the display of relative addresses (see below).

The following special address symbols are used to access registers:

$0 through $5 -- Registers R0 through R5.

$6 -- Register 6, the stack pointer (SP).

$7 -- Register 7, the program counter (PC).

$S -- Processor Status Word.

$M -- Mask register, used with data watchpoints (see below).

$F -- Print format control register. If the value in the $F register is 0 (zero) addresses are displayed in relative format (n,offset). If bit 0 (value 1) is set in the $F register, address operands of symbolic instructions are displayed as absolute rather than relative values. If bit 1 (value 2) is set, locations are displayed in absolute rather than relative form. If both bits (value 3) are set, all addresses are displayed in absolute form.

$B -- Start of instruction breakpoint address registers.

$R -- Start of relocation registers.

All values are assumed to be octal unless a decimal point terminates the value.

The data watchpoint facility is a powerful tool for determining where a particular data cell is being modified. The form of the command used to initiate a watchpoint is:

address;valueM

where address is the address of the word that is to be monitored, and value is an optional parameter that specifies a value to be watched for. If the value parameter is not specified, a data watchpoint occurs any time the value of the word changes. If a value parameter is specified, a watchpoint only occurs when the value of the word matches the specified value. When a watchpoint occurs, the program counter is pointing past the instruction that modified the word.

A data mask may be specified to select a portion of the word being monitored with the watchpoint. The mask value may be referenced using the register name "$M". Initially the value of the mask is 177777 which causes the entire word contents to be monitored. If some other value is stored in $M only the bits selected by the mask are monitored.

The data watchpoint facility causes the program to execute very slowly. The reason for this is that when a watchpoint is in effect the debugger causes a break to occur on each instruction executed and then checks to see if the monitored data word has changed value. If a watchpoint is in

effect but no instruction breakpoints are set, the execution of the program is slowed down by a factor of about 55 (i.e., the program takes 55 times as long to execute). If a watchpoint is in effect and instruction breakpoints are also set, the speed reduction factor is about 80. The best strategy is to set an instruction breakpoint as close as possible to the code that is modifying the data word and then when that breakpoint is reached, remove the instruction breakpoint, set the data watchpoint, and proceed with execution. There is no significant speed reduction when instruction breakpoints are set but no watchpoint is in effect.

The "[" and "]" operators are used to decode values into symbolic instruction form. The "[" operator can be used to decode a value that has just been displayed using the "/" operator or to display the contents of a specific location in symbolic form. The following example commands illustrate how this might be done (the commands typed by the user are underlined):

```
DBG:2030/010246  [  MOV     R2,-(SP)
DBG:3000[     BIT     #2000,[3,100]
```

The "]" operator closes the currently open cell, opens the next cell and displays its contents in symbolic instruction form. The number of bytes to skip from the currently open cell to the next cell is determined by the number of bytes used by the instruction in the currently open cell. Thus the "]" operator is convenient for examining a set of consecutive instructions.

If the debugging facility is wanted, the DBGFLG sysgen parameter in TSGEN must be set to 1 (one) when the system is generated. The debugging facility adds approximately 6.2Kb of code to the mapped portion of the system if it is included.

13. Keyboard commands and system service calls have been added to allow a user to allocate a device for exclusive access. Once a device has been allocated to a job, no other job is allowed to access the device. A device allocated to a primary line or associated virtual line may be accessed by any of the other virtual lines or the primary line. The form of the ALLOCATE command is:

    ALLOCATE device,device,... [logical_name]

    where "device" is the name of a physical device or logical device name that has been assigned to a physical device, and "logical_name" is an optional logical device name which, if specified, is assigned to the first physical device allocated. For example the following command allocates device MT0:

    ALLOCATE MT0

The following example allocates CL1 and DX0 and assigns the logical name LX to CL1:

```
ALLOCATE CL1,DX0 LX
```

The maximum number of devices that can be allocated simultaneously by all users is controlled by the MAXALC sysgen parameter. The TT, LD, and SY devices may not be allocated.

A SHOW ALLOCATE keyboard command is available to display a list of all allocated devices and their associated jobs. Typing "ALLOCATE" without any devices is equivalent to SHOW ALLOCATE.

The DEALLOCATE command is used to deallocate devices. The form of this command is:

```
DEALLOCATE device,device,...
```

or

```
DEALLOCATE/ALL
```

The first form of the command is used to deallocate specific devices. The second form is used to deallocate all devices allocated to the current job. The DEALLOCATE command can also be specified with no devices or options to deallocate all devices for the job (equivalent to DEALLOCATE/ALL).

Devices allocated by other users may not be deallocated. Devices may only be deallocated by the user who allocated them.

All devices allocated to a job are automatically deallocated by the system when the job logs off. Devices allocated by a primary line or virtual line are not deallocated by a request to deallocate all devices issued from a different virtual line. However, a request to deallocate a specific device will take effect if the device was allocated by the primary line or any of the virtual lines associated with the primary line that issues the deallocation request.

Three system service calls (EMT's) are available to control device allocations. All EMT's are of the form:

```
EMT     375
```

with R0 pointing to one of the argument blocks described below. The EMT argument block used to allocate a device is:

```
.BYTE   0,156
.WORD   dev_ptr
```

where "dev_ptr" is the address of a four word block which contains the device name in RAD50 form in the first word and zeros in the other three words. If an error is detected by the EMT, the carry-flag is set on return and the following error codes are returned:

Code	Meaning
1	Device has been allocated to another user
2	Invalid device specified
3	Device allocation table is full (Regenerate TSX-Plus and increase the MAXALC parameter)
4	Device is currently being used by another user

If the device has been allocated to or is in use by another job (error codes 1 and 4), then on return from the EMT R0 contains the number of that other job.

The form of the argument block used to deallocate a device is:

```
.BYTE     1,156
.WORD     dev_ptr
```

where "dev_ptr" is the address of a four word block which contains the device name in RAD50 form in the first word and zeros in the other three words. If an error is detected by the EMT, the carry-flag is set on return and the following error codes are returned:

Code	Meaning
1	Device has been allocated to another user
2	Invalid device specified

If the device has been allocated to another user (error code 1), on return from the EMT R0 contains the number of the job to which the device has been allocated.

The third EMT is used to check to see if a device has been allocated to another user. The form of this EMT argument block is:

```
.BYTE     2,156
.WORD     dev_ptr
```

where "dev_ptr" is the address of a four word block which contains the device name in RAD50 form in the first word and zeros in the other three words. If an error is detected by the EMT, the carry-flag is set on return and the following error codes are returned:

Code	Meaning
1	Device has been allocated to another user
2	Invalid device specified
4	Device is currently being used by another user

If the device has been allocated to or is in use by another job (error codes 1 and 4), then on return from the EMT R0 contains the number of that other job.

14. A new command, LOGOFF, has been implemented to log off a time-sharing line. It is synonymous with the OFF, BYE and KJOB commands.

15. Several enhancements have been made to the SHOW command. A SHOW SPOOL command is now available to display the names of the spooled devices. A SHOW ALLOCATE command may be used to display the names of allocated devices. A SHOW CL command displays information about CL (Communication Line) units. The SHOW TERMINALS command has been enhanced to display additional information about terminal lines including information about which lines are in use as CL (communication line) units. The following new terminal types are also displayed: "PC" = Professional console, "PP" = Professional printer port, "CP" = Professional communications port, "DH" = DH11 multiplexer lines, and "DHV" = DHV11 multiplexer lines. The name declared during system generation for identification of each line is also displayed. (See the description of the NAME macro.)

16. A logical device name may now be used with SET, SPOOL, and ACCESS commands as well as physical device names. For example, the following commands have the effect of applying the LENGTH=66 option to device CL1:

```
.ASSIGN CL1 XX
.SET XX LENGTH=66
```

The assignment must associate the logical name with a physical device. Pseudo-device names such as TERMINAL, TT, SL, CACHE, UCL, QUAN1, PRIORITY, etc. may not be accessed using logical device names.

17. A SET [NO]VERIFY keyboard command has been implemented to provide compatibility with the VMS operating system. This command has the inverse function of the SET TT [NO]QUIET command. That is:

```
SET VERIFY   =  SET TT NOQUIET
SET NOVERIFY =  SET TT QUIET
```

18. An enhancement has been made to the TSX-Plus terminal handler to allow it to accept high speed input with reduced risk of a "No free fork blocks" system error.

In version 5.1 of TSX-Plus, a software "silo" has been implemented for each line. The silo is used to hold characters that arrive in a burst before they are processed and moved to the terminal character input buffer. There are three sysgen parameters which set default parameters for the character silos. The NCSILO parameter specifies the default silo size. A value of 32 (decimal) is a reasonable silo size for most applications although you may want to use larger silos on lines that receive high-speed input from external devices or other computers. The NCXOFF parameter controls the system's attempts to prevent data input

overrun. When the silo fills to the point that only NCXOFF free character positions remain, the system will transmit an XOFF (ctrl-S) to try to stop transmission from the external device. If the system sends an XOFF because a silo becomes nearly full, it will send an XON (ctrl-Q) to restart transmission when there are only NCXON characters remaining in the SILO. The recommended values for NCXOFF and NCXON are 8 and 4 respectively.

The NCSILO, NCXOFF, and NCXON parameters specify default values which will apply to all lines. The SILO macro may be used within a line definition block to specify values other than the default for a specific line. The form of the SILO macro is:

```
SILO ncsilo,ncxoff,ncxon
```

The three parameters specify the silo size, XOFF point, and XON point for the line. The SILO macro must be used within a line definition block (between LINDEF and LINEND or between CLDEF and CLEND).

19. A new system generation macro, "SPEED", has been introduced as a replacement for the LINPRM macro. The SPEED macro can be used within a line definition block to specify the initial transmit/receive baud rate for a line. The form of the SPEED macro is:

```
SPEED     speed_code
```

Where "speed_code" must be one of the following symbolic names: S75, S110, S134.5, S150, S300, S600, S1200, S1800, S2000, S2400, S3600, S4800, S7200, S9600, S19200. That is, the speed code is formed by concatenating the letter "S" with the baud rate value.

The SPEED macro should be used within the line definition blocks for all lines that are connected to device controllers that have programmable baud rates except lines that are specified to have automatic baud rate selection ($AUTO flag).

The number of stop bits and even/odd parity setting that were previously provided with the LINPRM macro are no longer specified. The system now automatically uses 2 stop bits for speeds of 110 baud or slower and 1 stop bit for speeds faster than 110. All character transmission is now done using 8 bit codes without parity.

20. A new system generation macro called NAME may now be used within line definition blocks to specify a descriptive name for serial ports. The use of the NAME macro is optional. If it is used, it must appear between the LINDEF and LINEND macros for time-sharing lines or the CLDEF and CLEND macros for dedicated CL lines. The form of the NAME macro is:

NAME <descriptive text>

The descriptive text string may be up to 12 characters long and should be enclosed in angle brackets ("<" and ">"). The descriptive text has no effect on the operation of the system except that it is printed in the "Line Name" column of the SHOW TERMINALS command display and is shown in the SYSMON screen for terminal lines. The following example illustrates a line definition block using the NAME macro:

```
LINDEF  60,177560
NAME    <Console line>
TRMTYP  VT100
LINEND
```

21. A number of new options are available for use when defining a device by use of the DEVDEF macro in TSGEN. The new form of the DEVDEF macro is:

    DEVDEF <device>[,option,...,option]

    For each device to be available to the system an entry must be made using the DEVDEF macro. (Note that TT, LD, SL and CL must never be included in the list of device definitions.) The DEVDEF macro requires at least one argument, but may have several optional arguments as described below:

    1. The first parameter is the two character device name enclosed in angle brackets.

    2. The optional parameters specify the device characteristics. There are nine allowable device attributes which may be specified in any order. They are as follows:

       DMA Device performs Direct Memory Access (DMA).

       MAPIO Perform I/O mapping. This option must be specified for devices that perform DMA I/O and which have controllers or handlers which only support 18-bit addresses. This argument is only meaningful on the Q-bus.

       EVNBUF Require even byte buffer address for I/O transfers. An error will be generated if an I/O operation is directed to this device and the buffer is not on a word boundary.

       NOCACHE Do not use generalized data cache for this device.

       NOMOUNT Do not allow mounts (directory caching) for this device.

       REQALC Require device allocation before use. If this option is specified, users are not allowed to access units on the device unless the units have been allocated for their exclusive use with the ALLOCATE command or the EMT to allocate the device

MAPH Load the device handler outside the low memory 40K byte region and into a mapped handler region. (See below).

NOMAPH Do not load the handler into a mapped handler region; always load it into the low memory 40k byte region.

HANBUF Handler contains an internal I/O buffer which is used for DMA transfers.

In version 5.1 of TSX-Plus, some device handlers may be placed in extended memory regions reducing the size of the system in the critical 40Kb low memory area. Handlers which are placed in extended memory are known as "mapped" handlers. In order to be eligible for mapping, a handler must meet the following rules:

1. The handler must not be larger than 8Kb.

2. The handler must not use Page Address Register (PAR) 5. Mapped handlers are accessed by mapping kernel PAR 5 to the handler while they are running.

3. The handler may not connect to more than two device interrupt vectors and the addresses of the vectors must be specified in the handler header (the standard way). Handlers may not dynamically connect to interrupt vectors after they have been loaded because the system must generate an interrupt intercept to cause PAR 5 to be mapped to the handler before the interrupt is passed to the handler.

4. If the handler performs DMA I/O operations to or from a buffer area located within the handler, the following additional restrictions apply:

   a. The current position of the handler in physical memory must be taken into account when computing the physical address of the buffer. (See example below.)

   b. Handlers that do DMA I/O to internal buffers may not be mapped on UNIBUS systems with more than 256Kb of memory (systems using UNIBUS mapping).

   c. Handlers that do DMA I/O to internal buffers may not be mapped on QBUS systems with more than 256Kb if the device controllers do not support 22-bit addresses (i.e., if the MAPIO option must be specified).

Restrictions "b" and "c" above are automatically checked by the system and the handler is loaded in low memory if they are not met.

The HANBUF option is used with the DEVDEF macro to declare which handlers perform DMA transfers to internal buffers. This flag is automatically set as appropriate for standard handlers such as MT and need not be specified with the DEVDEF for standard supplied handlers.

The following subroutine illustrates how a handler can translate the virtual address of an internal I/O buffer into the correct physical address. This subroutine will function correctly whether or not the handler is mapped:

```
;  Calculate a 22-bit buffer address from the virtual address
;  of a buffer which is contained within the device handler.
;  This is necessary if the handler is mapped.
;
;  Input:
;     OLDBA - Virtual address of the internal buffer
;     EXTADR - Zero
;
;  Output:
;      OLDBA - Low order (16-bits) of the physical address
;      EXTADR - High order (6-bits) of the physical address
;      All registers are preserved
;
MAPADR: CMP     OLDBA,#120000     ;Handler loaded into a mapped region?
        BLO     10$               ;Br if handler is below mapped region
        MOV     R0,-(SP)          ;Save registers
        MOV     R1,-(SP)          ;
        MOV     @#172352,R1       ;Get the handler PAR5 relocation value
        CLR     R0                ;Clear high order address cell
        ASHC    #6,R0             ;Convert to physical memory address
        BIC     #160000,OLDBA     ;Isolate offset (clear virtual PAR #)
        ADD     R1,OLDBA          ;Add the phys. relocation to low order
        ADC     R0                ;Add carry to high order
        ASH     #4,R0             ;Shift high order to bits 4 thru 9
        MOV     R0,EXTADR         ;Store high order address
        MOV     (SP)+,R1          ;Restore registers
        MOV     (SP)+,R0          ;
10$:    RETURN                    ;Return
```

The following default flags are set by the system for standard devices:

Device	Options
CR	MAPH
CT	MAPH
DD	NOMAPH
DL	DMA,MAPH,HANBUF
DM	DMA,NOMAPH
DP	DMA
DS	DMA
DT	DMA
DU	DMA
DW	MAPH
DX	MAPH
DY	DMA,NOMAPH
DZ	MAPH
FW	DMA
LP	MAPH
LS	MAPH
MM	DMA,MAPH,HANBUF
MS	DMA,MAPH,HANBUF
MT	DMA,MAPH,HANBUF
NL	MAPH
PC	MAPH
RF	DMA
RK	DMA,MAPH
VM	EVNBUF,NOCACHE,NOMAPH
XC	MAPH
XL	MAPH

If neither the MAPH nor the NOMAPH options is specified for a handler, the handler is not mapped. If both the NOMAPH and MAPH options are specified, the handler is not mapped. Thus it is possible to override the default MAPH option for a standard handler by specifying NOMAPH.

It is no longer necessary to have device handlers installed in RT-11 when TSX-Plus is started. The only device handlers that must be installed are handlers for the devices where the TSX-Plus swap and spool files are placed.

TSX-Plus will now refuse to install new device handlers which were issued by Digital subsequent to the version of RT-11 under which TSX-Plus is being started. In other words you must upgrade to the appropriate version of RT-11 in order to be able to use the newer device drivers with TSX-Plus. Specifically, the following device drivers require the indicated version of RT-11:

Device	RT-11 version
DU	5.00
XL	5.01

22. Real-time support is now always included in the system regardless of the value specified for the RTVECT system generation parameter. RTVECT must still be set to a non-zero value if real-time interrupt vectors are to be connected to TSX-Plus programs but the basic real-time support facility including the ability to map to the I/O page are now always included in the system.

23. The distribution contains source language patch files which must first be applied to the appropriate RT-11 driver source when rebuilding handlers. See the TSX-Plus System Manager's Guide for information on building device handlers. Generally it is not necessary to rebuild device drivers as they are supplied for standard configurations. Source language patch files are provided for RT-11 versions 5.00 and 5.01 and are named "ddxxx.SLP" where "dd" represents the two character device driver name and "xxx" represents the RT-11 version number. If it is necessary to rebuild a device driver, the following additional definition must be included in TSXCND:

    TSX$P = 1

    This conditional is used to trigger the inclusion of TSX-Plus specific code.

24. Device driver installation code is now executed during system initialization. All registers are preserved for use by the device installation procedure. If the C-flag is set on return from the handler installation code to the system, the handler is not installed. Note that handlers no longer have to be installed in RT-11 to be installed in TSX-Plus.

25. The SET command can now be used to set options in handlers for devices which are not installed in TSX-Plus. When this is done the set is applied to the handler disk file (i.e., SY:xx.TSX) and the following warning message is printed:

    ?KMON-W-Handler not installed. Set applied to handler disk file only.

26. An improvement has been made in the handling of .FORK requests for interrupts. Under TSX-Plus version 5.1, fork requests have associated processing priorities and a higher priority fork request may interrupt the processing of a lower priority fork.

    In order to understand the processing of fork requests by TSX-Plus it is helpful to understand the concept of interrupt priorities. The PDP-11 family of computers have 8 interrupt priority levels, numbered 0 through 7. The priority of an interrupt is selected by the device requesting the interrupt, and the processor (CPU) remembers the current interrupt priority in the processor status word. An interrupt request is held in a pending state and is not allowed to interrupt the processor if the current interrupt priority is equal to or greater than the pending interrupt priority. Priority level 0 is the priority at which the processor runs when no interrupt is being serviced.

    Fork processing under TSX-Plus implements a software based interrupt system which effectively operates at an interrupt priority level greater than hardware priority 0 and less than hardware priority 1. Like the hardware interrupt system, TSX-Plus fork requests have priority values that are specified by the software component that is making the fork request. Also like the hardware interrupt system, a fork request may interrupt a currently executing fork request of lower priority but may not interrupt a currently executing fork request of equal or greater priority. The fork priority values range from 1 to 127 (decimal); the higher the numerical value, the higher the priority.

    Conceptually you can think of the fork priorities as corresponding to interrupt priorities in the range 0.001 through 0.127. Thus, any hardware interrupt request (which has a priority in the range 1 through 7) can interrupt any fork request. Fork requests are queued in order by priority value. If two or more requests have the same priority value, they are queued in the order in which the requests were made.

    The standard instructions generated by a .FORK macro are:

    ```
    JSR     R5,@$FKPTR
    .WORD   fkblk-.
    ```

    Where $FKPTR is a cell containing the address of the system fork routine and fkblk is the address of a 4-word fork block. This fork call works under both RT-11 and TSX-Plus. If this form of fork call is used under TSX-Plus, the fork request is queued with a fork priority of 50 (decimal).

An alternative fork request call may be used under TSX-Plus to specify a priority for the fork. The form of this call is:

```
JSR     R5,@$FKPTR
.WORD   100000+priority
```

Where "priority" is a fork priority value in the range 1 to 127. The constant 100000 must be added to the priority value to produce a value which TSX-Plus can recognize as a priority rather than the offset to a user fork block. Note that, since TSX-Plus uses internal fork request blocks, the second fork request form does not specify the offset of a user-supplied fork block.

Current TSX-Plus fork priority values are defined in the initial part of TSGEN with symbols that start with "FP$".

The use of fork priorities reduces the latency time between the occurrence of an interrupt and the time that a real-time interrupt service routine is entered. It also reduces the likelihood of having a "No free fork blocks" system error.

27. Support is now included for the IB device handler which controls the DEC IB(V)11 interface card for IEEE GPIB devices. This required the addition of an executive routine ($RELOC) to TSX-Plus which relocates a user's virtual address to a PAR6 mapping value and boundary. To interface the IB routines to TSX-Plus, changes are required to both TSGEN and the IB handler.

    The change in TSGEN is necessary since the normal IB subroutines attempt to open the IB device on decimal channel numbers 16, 17, 18, and 19. Increasing the TSGEN parameter "NUCHN" from 16 to 20 (decimal) allocates additional channels and allows the IB subroutines to execute without changes. (It may be necessary to allocate more channels in non-standard configurations.) The change in the IB handler is due to the use of PAR6 for mapping to the user buffer by TSX-Plus rather than PAR1 as in RT-11. This requires the following three changes to the IB handler:

		old value	new value
PAR1	=	172342	172354
PAR1HI	=	37776	157776
PAR1LO	=	20000	140000

28. A facility has been added to allow one job to monitor the status of one or more other jobs. Once a job has declared that it wishes to monitor another job, a completion routine in the monitoring job is entered whenever a status change occurs for the job being monitored.

This facility is useful in situations where several jobs are cooperating with each other. For example, one job might act as a database server for several other jobs. If one of the jobs being served aborts, the database server needs to know so that it can unlock records held by the aborted job and possibly close channels or files.

There are three system services associated with the job monitoring facility: (1) establish a monitoring connection with another job; (2) cancel a monitoring connection; and (3) broadcast a status message to monitoring jobs.

The form of the EMT used to establish a monitoring connection between two jobs is:

```
EMT     375
```

with R0 pointing to the following argument block:

```
.BYTE   0,157
.WORD   job_number
.WORD   cpl_rtn
```

where "job_number" is the number of the job to be monitored, and "cpl_rtn" is the address of a completion routine to be entered when the status of the job changes. The following errors can be reported by this EMT:

Code	Meaning
1	Invalid job number specified
2	No free job monitoring control blocks

If error 2 occurs (no free job monitoring control blocks), increase the value of the MAXMON sysgen parameter.

The specified completion routine will be entered each time the status of the monitored job changes or the monitored job executes the EMT to broadcast a status message (see below). On entry to the completion routine, the low-order byte of R0 contains the number of the job whose status change is being reported. The high-order bit (100000) of R0 is 0 (zero) if the status report was generated by the system and is 1 (one) if the status report was generated by a program executing the EMT described below to broadcast a status message. R1 contains a 16-bit status value. The system generated status values are listed below:

Value	Meaning
1	Job has been initialized
2	Job has logged on using LOGON program
3	Job began running a program
4	Job returned control to the keyboard monitor
5	Job logged off

Status code 1 is reported when a line is started after it has been logged off. This normally occurs when a carriage return is received by a logged-off line. Status code 2 occurs after a job has successfully logged in using the LOGON program. Jobs which do not invoke the LOGON program will not generate status value 2. Status code 3 is generated when the keyboard monitor transfers control to a program whose execution is being started. Status code 4 is generated when a program exits or aborts and control is returned to the keyboard monitor. If one program chains to another program status code 3 is generated when the second program is started but status code 4 is not generated since the chain does not return control to the keyboard monitor. Status code 5 is generated when a job logs off.

The form of the EMT used to cancel the monitoring of a job has the form:

```
EMT     375
```

with R0 pointing to the following argument block:

```
.BYTE   1,157
.WORD   job_number
```

Where "job_number" is the number of the job for which monitoring is to be cancelled. If a job number of 0 (zero) is specified, all job monitoring requests for the job issuing the EMT are cancelled. An error code of 1 is returned by this EMT if an invalid job number is specified.

All job monitoring requests issued by a job are cancelled by the system when the job exits, chains, or does a .SRESET or .HRESET.

The form of the EMT used to broadcast a status change value is:

```
EMT     375
```

with R0 pointing to the following argument block:

```
.BYTE   2,157
.WORD   status_value
```

where "status_value" is a 16-bit status code value that is to be broadcast to all jobs that are monitoring the job which issues this EMT. On entry to the completion routines in the monitoring job(s), the status value is in R1, the number of the job issuing the EMT is the low-order byte of R0, and the high-order bit (100000) of R0 is 1 (one).

If there are no jobs monitoring the job issuing this EMT, the EMT will generate an error with a status code of 0 (zero).

29. A new EMT has been added to allow a program to post a "read" request for a TSX-Plus inter-job message and to specify a completion routine to be called by the system when a message arrives. This allows a program to continue executing while waiting for a message to arrive. The form of the EMT is:

```
EMT     375
```

with R0 pointing to the following argument block:

```
.BYTE   1,106
.WORD   chadr
.WORD   msadr
.WORD   mssiz
.WORD   cplrtn
```

where "chadr" points to a six byte field containing the message channel name, "msadr" points to the buffer where the message is to be placed, "mssiz" is the size of the message buffer (bytes), and "cplrtn" is the address of the completion routine to be called asynchronously by the system when a message arrives.

On entry to the completion routine, R0 contains the number of bytes in the message that was received, and R1 contains the number of the job that sent the message. Another EMT of this form may be issued from within the completion routine if it is desired to receive additional messages.

The following error codes can be returned by this EMT:

Code	Meaning
1	All message channels are busy (increase MAXMC).
5	Max message requests are active (increase MAXMRB).

The .SPND and .RSUM system service calls may be used with this facility if the program reaches a point where it must suspend its execution until a message arrives.

A new sysgen parameter related to message communication has been added to TSGEN. The new parameter named MAXMRB specifies the maximum number of requests for messages that may be stored by the system at any time. This count must include requests that specify completion routines, requests that test for messages, and requests that wait for messages. If a request for a message is made and the maximum number of requests are already pending, an error code of 5 is returned from the EMT.

30. The spooling system has been enhanced to allow deletion of spool files that are pending in the spool queue. The new form of the SPOOL delete command is:

    SPOOL dev,DELETE [id]

    where dev is the name of a spooled device and id is an optional file identification number used to specify which file in the spool queue is to be deleted. The SHOW QUEUE command has been changed to list file identification (ID) numbers along with other spool file information. An example of the output produced by the SHOW QUEUE command is shown below:

ID	Dev	Job	File	Form	Blocks
37	LP *	1	PAYROL	STD	135
39	LP	1	GL	STD	25
41	LP	1	AR	STD	86

    The "ID" column provides a unique identification number for each spool file.

    When an identification number is specified with the SPOOL DELETE command, that number identifies the file to be deleted. In this case the spooled device (dev) may be the name of any spooled device and does not have to match the device for which the file is pending.

    If the spool file identification number is omitted, the SPOOL DELETE command deletes the file which is currently being printed on the specified device.

    If the specified file is currently open, the file is marked to be deleted but is not actually deleted until it is closed and the spooler begins to process it. If the file is currently being printed when the delete command is issued, the spooler deletes blocks as they become available and the spooled device is tied up until the file deletion is completed.

    Logical device names may now be used with the SPOOL command. For example, if CL2 is a spooled device and the logical name LP is assigned to CL2 by use of a command of the form:

    .ASSIGN CL2 LP

    then the following commands are both legal and are equivalent:

    .SPOOL CL2,STAT
    .SPOOL LP,STAT

    A change has been made to the way that a spooled device name is recognized. Previously if a device name such as LP, without a unit number, was specified as a spooled device, the device name LP0 would not be recognized as a spooled device but would directly access the printer, bypassing the

spooling system. In version 5.1, the names LP and LPO are equivalent and are both recognized as accessing the printer through the spooling system.

It is now legal to issue .READ and .SPFUN requests to a spooled device. These operations bypass the spooling system and directly connect with the device handler.

31. The size of the portion of the system which goes in the 40Kb low memory area has been reduced. It may now be possible in some cases to start TSX-Plus under RT-11/FB as well as RT-11/SJ.

32. A new option is available for the RUN and R commands to cause the PAR 7 region of the program being started to be mapped to the I/O page rather than the simulated RMON. The name of the new option is "/IOPAGE". Operator privilege is required to start a program with this option. For example, the following command starts a program named GRAPH1 and maps PAR 7 to the I/O page:

```
RUN/IOPAGE GRAPH1
```

33. A change has been made to allow access to all devices and files while a start-up command file is executing. Thus an ACCESS command placed in a start-up command file does not take effect until the end of the start-up command file is reached.

34. TSXUCL has been modified to improve job start-up. When several user-defined commands are defined in a start-up command file, TSXUCL is called separately to define each command. In order to speed up multiple command definitions, TSXUCL may be run directly from the start-up command file, followed by a list of command definitions. TSXUCL attempts to create a user-defined command for each successive input line until it finds a blank line. Using this method, it only returns to KMON after finding a blank input line. This method bypasses the repeated calling and returning from TSXUCL and is consequently much faster if many user-defined commands are included in the start-up command file. Equivalent start-up command files in each style might look like:

```
Slower                  Faster
------                  ------
!Slower method          R TSXUCL !Faster method
Q :== SPOOL LP,STAT     Q :== SPOOL LP,STAT
WHEN :== SHOW TIME      WHEN :== SHOW TIME
NEW :== DIR ^/D         NEW :== DIR ^/D
!End of command file    !Blank line required
```

35. The following changes and enhancements have been made to the SYSMON system monitoring program.

    1. The user times display has been modified so that PPN's are formatted on each cycle through the screen. Formerly, all PPN's were initially formatted in a table to lower processing time. However, this led to confusion if a line logged off and back on using a different account between two display cycles.

    2. The screen header has been redesigned and no longer includes the site name and license number.

    3. The default cycle time has been reduced to five seconds.

    4. The CL Device Display has been added. This display shows SET and status information on CL devices.

    5. XOFF received/sent information has been added to the Terminal Display. This information is convenient for debugging communications problems.

    6. SYSMON will now check for VT2xx type terminals when the line is set as a non-DEC terminal. Formerly, SYSMON only checked for VT52 and VT1xx type terminals.

    7. The Directory Cache Display now shows a maximum of 68 directory cache entries. This solves a problem encountered when a large number of directory cache entries were enabled in TSGEN, but not actually allocated.

    8. The Terminal Display will now cycle through terminal lines if zero is entered as a line number.

36. Logon accounts are no longer required to have passwords (although they are recommended). To logon to an account that does not have an associated password, press RETURN when the "Password:" prompt appears.

Corrected Problems

37. The following corrections are included in TSX-Plus version 5.1:

37.1 A problem has been corrected which occasionally caused TSX-Plus to lose completion routine requests. This problem only occurred when more than one completion routine was pending for the same job. The most common symptom was that the VTCOM program would sporadically die while running under TSX-Plus.

37.2 A problem was corrected which caused echoing of carriage return and line feeds when echoing had been disabled with the "F" program controlled terminal option. This occurred when the single line editor was set both "ON" and "TTYIN".

37.3 A problem was corrected which caused the error message "?MON-F-Invalid address as EMT argument" for valid I/O requests. The message resulted from checking the buffer address for a word (even address) boundary on all transfer requests. The corrected implementation is boundary checking only for devices which require word alignment such as cached devices, VM, and others.

37.4 A problem was corrected which occasionally resulted in corruption of the spooling system when message channels were included. An incorrect amount of memory had been reserved for message buffers.

37.5 A problem was corrected which permitted mapped overlay regions with unusually large option selections to overflow their 8Kb limit and corrupt portions of the system. This caused such errors as unexplainable shared file record locking failures.

37.6 A problem was corrected in VM which corrupted a memory location and caused unusual system errors when writing partial block transfers.

37.7 A problem was corrected so that the appropriate error level is passed to IND when invalid dates or times are entered.

37.8 The TSAUTH program has been corrected so that it is no longer possible to create duplicate PPN entries.

37.9 A problem related to non-terminating .GTLIN requests (bit 3 set in the JSW) has been corrected. This problem could cause the system to unexpectedly resume execution of a command file that was terminated by typing control-C while a non-terminating .GTLIN EMT was accepting input from the terminal.

37.10 The EMT trace facility now lists invalid EMT's and EMT's with invalid channel numbers.

37.11 A problem has been corrected so that an inner command file no longer aborts an outer command file through an intermediate IND control file.

37.12 A problem has been corrected which occasionally caused lines to be echoed twice if the NOWAIT terminal control option was used.

37.13 An escape character now echoes as "$" (dollar sign) when in non-special TTY input mode and SL is not active. Note that this prevents directly entering screen control sequences from the keyboard.

37.14 If VT100 escape sequences are declared to be an activation condition and an escape sequence is received before the program declares an input field width limit, the escape sequence is allowed to be accepted in its entirety.

37.15 The system uptime portion of the SYSTAT display has been corrected. Previously, when the uptime exceeded 24 hours, the minute field was always 32.

37.16 Three CCL commands were corrected. The FORTRAN/SHOW switch no longer requires an argument. The DUMP/FOREIGN switch has a spelling mistake corrected. The EDIT command has been altered to operate identically to RT-11 when invoking either EDIT or TECO.

37.17 The .READ, .READW, and .READC EMT now checks the size of read requests to insure that the word count is less than 32,767. This differs from the RT-11 implementation which always queues the read request to the device handler. Device handlers interpret a read request with a word count above 32,767 as a write.

37.18 A correction has been made to the command processor which caused a problem with command line recognition. An example of the problem was the command "SY:DIR" which executed the SYSTAT command rather than executing the DIR command file or program on SY:.

37.19 A rare problem was corrected which caused a kernel mode trap when attempting to squeeze a device which had previously active generalized cache entries after disabling cache.

37.20 A correction was made which echoed terminating carriage returns and line feeds when SL was set ON and TTYIN and the terminal option "F" (turn off character echoing) was invoked.

37.21 A problem has been corrected which sometimes caused the KILL command to be ineffective if the job being killed was suspended due to its terminal output buffer being full because control-S (XOFF) was received.

37.22 The total size of installed memory displayed with the MEMORY command now reports the total memory seen by the initialization code. This may be more than the amount of memory used by TSX-Plus if the MEMSIZ parameter is used to allocate less than the total. Machines which have 18-bit backplanes and full memory (256 K bytes) will now size correctly as long as EXTMCH is correctly set to zero (0).

37.23 Control-O suppresses terminal echoing during command file execution.

37.24 A rare problem in the I/O completion code which caused system failure when using two magtape devices on one controller has been corrected.

37.25 Checks were added to the EMT which sets program priority so that only valid priorities are accepted.

37.26 The SPOOL LP,ALIGN,filnam command was corrected to automatically convert the file name to upper case to prevent a KMON trap.

37.27 Form mount requests within a file are no longer case sensitive and are converted internally to upper case.

37.28 Both privileged and non-privileged accounts are allowed access to all devices and files during the startup command file. This corrects a discrepancy which existed between the execution of startup command files for privileged and non-privileged accounts which actually denied access to files for privileged accounts unless specific access was granted (by use of the ACCESS command). This does not affect access handling once the startup command file has completed.

37.29 A problem has been corrected which could cause a job to become hung in I/O wait state if the job aborted at a critical time while performing I/O to a device with generalized data caching in effect.

37.30 An improvement has been made in the technique used to stop I/O intensive jobs which need to be swapped out of memory. The new technique is more effective for jobs that do continuous I/O and use I/O completion routines and corrects a problem which could result in a job being hung in I/O wait state in some cases if a swap was attempted while lengthy I/O operations were in progress.

37.31 A problem has been corrected which could cause a shared file record locking request to be blocked by a request issued by another job to the same logical block number on a different logical disk on the same physical disk.

INDEX